

CLAIMS

What is claimed is:

1 1. A method for displaying a structural view of a computer program during a  
2 debugging session, said method applying within a computer system, said method  
3 comprising:

4 displaying a portion of a program call graph (PCG), wherein said PCG  
5 includes a P\_node symbolically representing a first procedure and a procedure  
6 relationship symbolically representing a calling association from said first procedure  
7 to a second procedure.

1 2. The method of claim 1, further comprising:

2 determining a condition for said first procedure while executing the computer  
3 program; and

4 marking said P\_node based on said condition into a marked P\_node, wherein  
5 said marked P\_node is visually distinguishable from said P\_node.

1 3. The method of claim 2, wherein said condition is taken from the group  
2 consisting of an execution state, an execution frequency and an execution age,  
3 wherein said execution state corresponds to either said first procedure having been  
4 executed or said first procedure being nonexecuted, wherein said execution frequency  
5 is a rate of said first procedure being executed, and wherein said execution age is a  
6 time interval since said first procedure has been executed.

1 4. The method of claim 2, wherein said marking is changing a symbolic attribute  
2 from an unaltered P\_node, wherein said symbolic attribute is taken from the group  
3 consisting of shade, highlight, color, border thickness, symbol size, symbol shape, and  
4 alternation of a visual characteristic.

1 5. The method of claim 1, further comprising:  
2 creating a list of a plurality of PCG procedures from said portion of said PCG;  
3 and  
4 recording said list of said plurality of PCG procedures onto a memory, said  
5 memory being retrievable.

1 6. A method for displaying a structural view of a computer program during a  
2 debugging session, said method applying within a computer system, said method  
3 comprising:  
4 displaying a portion of a control flow graph (CFG), wherein said CFG  
5 includes a B\_node symbolically representing a first basic block and a basic block  
6 relationship symbolically representing a calling association from said first basic block  
7 to a second basic block.

1 7. The method of claim 6, further comprising:  
2 displaying within said B\_node a line number associated with a source code  
3 statement of the computer program.

1 8. The method of claim 7, further comprising:  
2 displaying within said B\_node a portion of said source code statement.

1 9. The method of claim 8, wherein said portion of said source code statement can  
2 be alternately toggled for one of either displaying said source code statement or  
3 displaying said portion of said source code statement.

1 10. The method of claim 6, further comprising:  
2 determining a condition for said first basic block while executing the computer  
3 program; and  
4 marking said B\_node based on said condition into a marked B\_node, wherein  
5 said marked B\_node is visually distinguishable from said B\_node.

1 11. The method of claim 10, wherein said condition is taken from the group  
2 consisting of an execution state, an execution frequency and an execution age,  
3 wherein said execution state corresponds to either said first procedure having been  
4 executed or said first procedure being nonexecuted, wherein said execution frequency  
5 is a rate of said first procedure being executed, and wherein said execution age is a  
6 time interval since said first procedure has been executed.

1 12. The method of claim 10, wherein said marking is changing a symbolic  
2 attribute from an unaltered B\_node, wherein said symbolic attribute is taken from the  
3 group consisting of shade, highlight, color, border thickness, symbol size, symbol  
4 shape, and alternation of a visual characteristic.

1 13. The method of claim 6, further comprising:  
2 creating a list of a plurality of CFG instructions from said portion of said CFG;  
3 and

4 recording said list of said plurality of CFG instructions onto a memory, said  
5 memory being retrievable.

1 14. A programmable storage device readable by a machine tangibly embodying a  
2 program of instructions executable by said machine to perform method steps for  
3 displaying a structural view of a computer program during a debugging session, said  
4 program applying within a computer system, said method steps comprising:

5 displaying a portion of a program call graph (PCG), wherein said PCG  
6 includes a P\_node symbolically representing a first procedure and a procedure  
7 relationship symbolically representing a calling association from said first procedure  
8 to a second procedure.

1 15. The programmable storage device of claim 14, wherein said method steps  
2 further comprise:  
3 determining a condition for said first procedure while executing the computer  
4 program; and  
5 marking said P\_node based on said condition into a marked P\_node, wherein said  
6 marked P\_node is visually distinguishable from said P\_node.

1 16. The programmable storage device of claim 15, wherein said condition is taken  
2 from the group consisting of an execution state, an execution frequency and an  
3 execution age, wherein said execution state corresponds to either said first procedure  
4 having been executed or said first procedure being nonexecuted, wherein said  
5 execution frequency is a rate of said first procedure being executed, and wherein said  
6 execution age is a time interval since said first procedure has been executed.

1 17. The programmable storage device of claim 15, wherein said marking is  
2 changing a symbolic attribute from an unaltered P\_node, wherein said symbolic  
3 attribute is taken from the group consisting of shade, highlight, color, border  
4 thickness, symbol size, symbol shape, and alternation of a visual characteristic.

1 18. The programmable storage device of claim 14, wherein said method steps  
2 further comprise:

3 creating a list of a plurality of PCG procedures from said portion of said PCG;  
4 and  
5 recording said list of said plurality of PCG procedures onto a memory, said  
6 memory being retrievable.

1 19. A programmable storage device readable by a machine tangibly embodying a  
2 program of instructions executable by said machine to perform method steps for  
3 displaying a structural view of a computer program during a debugging session, said  
4 program applying within a computer system, said method steps comprising:

5 displaying a portion of a control flow graph (CFG), wherein said CFG  
6 includes a B\_node symbolically representing a first basic block and a basic block  
7 relationship symbolically representing a calling association from said first basic block  
8 to a second basic block.

1 20. The programmable storage device of claim 19, wherein said method steps  
2 further comprise:

3 displaying within said B\_node a line number associated with a source code  
4 statement of the computer program.

1 21. The programmable storage device of claim 19, wherein said method steps  
2 further comprise:

3 determining a condition for said first basic block while executing the computer  
4 program; and

5 marking said B\_node based on said condition into a marked B\_node, wherein  
6 said marked B\_node is visually distinguishable from said B\_node.

1 22. The programmable storage device of claim 21, wherein said condition is taken  
2 from the group consisting of an execution state, an execution frequency and an  
3 execution age, wherein said execution state corresponds to either said first procedure  
4 having been executed or said first procedure being nonexecuted, wherein said  
5 execution frequency is a rate of said first procedure being executed, and wherein said  
6 execution age is a time interval since said first procedure has been executed.

1 23. The programmable storage device of claim 22, wherein said marking is  
2 changing a symbolic attribute from an unaltered B\_node, wherein said symbolic  
3 attribute is taken from the group consisting of shade, highlight, color, border  
4 thickness, symbol size, symbol shape, and alternation of a visual characteristic.

1 24. The programmable storage device of claim 19, further comprising:  
2 creating a list of a plurality of CFG instructions from said portion of said CFG; and  
3 recording said list of said plurality of CFG instructions onto a memory, said memory  
4 being retrievable.

1 25. A debugger for displaying a structural view of a computer program during a  
2 debugging session, said debugger operating within a computer system, said debugger  
3 comprising:

4 a display for displaying a portion of a program call graph (PCG), wherein  
5 said PCG includes a P\_node symbolically representing a first procedure and a  
6 procedure relationship symbolically representing a calling association from said first  
7 procedure to a second procedure.

1 26. The debugger of claim 25, further comprising:

2 a condition determiner for determining a condition for said first procedure  
3 while executing the computer program; and  
4 a marker for marking said P\_node based on said condition into a marked  
5 P\_node, wherein said marked P\_node is visually distinguishable from said P\_node.

1 27. The debugger of claim 26, wherein said condition is taken from the group  
2 consisting of an execution state, an execution frequency and an execution age,  
3 wherein said execution state corresponds to either said first procedure having been  
4 executed or said first procedure being nonexecuted, wherein said execution frequency  
5 is a rate of said first procedure being executed, and wherein said execution age is a  
6 time interval since said first procedure has been executed.

1 28. The debugger of claim 26, wherein said marker changes a symbolic attribute  
2 from an unaltered P\_node, wherein said symbolic attribute is taken from the group  
3 consisting of shade, highlight, color, border thickness, symbol size, symbol shape, and  
4 alternation of a visual characteristic.

1 29. The debugger of claim 25, further comprising:  
2 a lister for producing a list of a plurality of PCG procedures from said portion  
3 of said PCG; and  
4 a recorder for recording said list of said plurality of PCG procedures onto a  
5 memory, said memory being retrievable.

1 30. A debugger for displaying a structural view of a computer program during a  
2 debugging session, said debugger operating within a computer system, said debugger  
3 comprising:

4 a display for displaying a portion of a control flow graph (CFG), wherein  
5 said CFG includes a B-node symbolically representing a first basic block and a basic  
6 block relationship symbolically representing a calling association from said first basic  
7 block to a second basic block.

1 31. The debugger of claim 30, further comprising:  
2 a line number display for displaying within said B-node a line number  
3 associated with a source code statement of the computer program.

1 32. The debugger of claim 31, further comprising:  
2 a statement display for displaying within said B-node a portion of said  
3 source code statement.

1 33. The debugger of claim 32, wherein said portion of said source code statement  
2 can be alternately toggled for one of either displaying said source code statement or  
3 displaying said portion of said source code statement.

1 34. The debugger of claim 30, further comprising:  
2       a condition determiner for determining a condition for said first basic block  
3 while executing the computer program; and  
4       a marker for marking said B\_node based on said condition into a marked  
5 B\_node, wherein said marked B\_node is visually distinguishable from said B\_node.

1 35. The debugger of claim 34, wherein said marker changes a symbolic attribute  
2 from an unaltered B\_node, wherein said symbolic attribute is taken from the group  
3 consisting of shade, highlight, color, border thickness, symbol size, symbol shape, and  
4 alternation of a visual characteristic.

1 36. The debugger of claim 34, wherein said condition is taken from the group  
2 consisting of an execution state, an execution frequency and an execution age,  
3 wherein said execution state corresponds to either said first procedure having been  
4 executed or said first procedure being nonexecuted, wherein said execution frequency  
5 is a rate of said first procedure being executed, and wherein said execution age is a  
6 time interval since said first procedure has been executed.

1 37. The debugger of claim 36, further comprising:  
2       a lister for producing a list of a plurality of CFG instructions from said portion  
3 of said CFG; and  
4       a recorder for recording said list of said plurality of CFG instructions onto a  
5 memory, said memory being retrievable.